UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/955,796	09/18/2001	Ed O. Schlotzhauer	10010804-01	1044

22878 7590 01/08/2009

AGILENT TECHNOLOGIES INC.
INTELLECTUAL PROPERTY ADMINISTRATION,LEGAL DEPT.
MS BLDG. E P.O. BOX 7599
LOVELAND, CO 80537

EXAMINER	
WEST, JEFFREY R	

ART UNIT	PAPER NUMBER
2857	

NOTIFICATION DATE	DELIVERY MODE
01/08/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

IPOPS.LEGAL@agilent.com

Office Action Summary	Application No.	Applicant(s)	
	09/955,796	SCHLOTZHAUER ET AL.	
	Examiner	Art Unit	
	Jeffrey R. West	2857	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *25 September 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-29 and 31-40* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-29 and 31-40* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *05 December 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

3. Claims 1-4, 7-9, 14-29, 31-33, and 36-40 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 6,401,220 to Grey et al.

With respect to claim 1, Grey discloses a method for a user of a measurement process to cause a variation in the measurement process (column 33, lines 38-46 and column 37, lines 10-19), the measurement process comprising a sequence of

operations controlled by a computer program (column 5, lines 53-57 and column 7, lines 18-30) containing a variation point at which a function call instruction is inserted by a designer of the computer program (column 20, lines 58-65) to pass control to a user-defined variation function (column 20, lines 58-65 and column 33, lines 38-46), said method comprising determining the variation to the measurement process (column 20, lines 58-65), providing a user-generated process modification software module comprising the user-defined variation function for causing the variation (column 20, lines 58-65, column 33, lines 38-46, and column 37, lines 10-19), and associating the function call instruction with the user-defined variation function (column 17, lines 25-29, column 18, lines 4-17, and column 20, lines 58-65), generating an executable variation of the measurement process (column 18, lines 4-17) wherein the function call instruction passes control to the user-defined variation function when the variation point in the computer program is reached (column 18, lines 4-17 and column 20, lines 58-65) and wherein the user is prevented from modifying the measurement process other than through the user-defined variation function (column 32, lines 14-16 and column 34, lines 15-40), said user being different from said designer (i.e. “sequence developer” different from the “user”) (column 20, lines 58-65).

With respect to claim 2, Grey discloses that the process modification software module further comprises an interface servicing element that services an interface realized by the measurement process (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

With respect to claim 3, Grey discloses that said interface operates in accordance with a predetermined protocol (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

With respect to claim 4, Grey discloses that said predetermined protocol is specified at a binary level (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

With respect to claim 7, Grey discloses that said interface has an identity which is determined by the user and said identity is identified and passed into said measurement process (column 12, lines 41-48, column 13, lines 50-62, and column 18, lines 4-16 and 28-37).

With respect to claim 8, Grey discloses that said process modification software module is one of a computer program conforming to a software component specification for distributed applications or dynamically linked library (i.e. C, C++, JAVA, Visual Basic) (column 13, lines 53-57 and column 14, lines 66-67).

With respect to claim 9, Grey discloses that the measurement process and the process modification software module are executed in a shared computer memory space (i.e. the test executive software performs the measurement and the measurement modification) (column 11, lines 41-56 and column 58, lines 60-67)

With respect to claim 14, Grey discloses that said variation comprises modification of data (column 15, lines 11-14).

With respect to claim 15, Grey discloses that said variation comprises modification of one or more numerical parameters of the measurement process (i.e. voltages) (column 30, lines 49-52 and column 46, lines 30-35).

With respect to claim 16, Grey discloses that said variation comprises modification of one or more control parameters of the measurement process, wherein one or more alternatives within the measurement process may be selected (column 19, lines 33-39 and column 21, lines 5-11).

With respect to claim 17, Grey discloses that said measurement process is applied to a device under test and said variation comprises alteration of a configuration of the device under test (column 18, lines 62-63 and column 19, lines 33-39).

With respect to claim 18, Grey discloses that said measurement process is applied to a device under test and said variation comprises causing input signals to be supplied to the device under test (column 10, line 62 to column 11, line 6 and column 19, line 64 to column 20, line 5).

With respect to claim 19, Grey discloses that said computer program contains a plurality of variation points and said process modification software module comprises a plurality of user-defined functions and wherein each of the plurality of variation points is associated with one of the plurality of user-defined functions (column 13, lines 16-25 and 32-44, column 14, lines 52-65, column 18, lines 28-37 and 49-54, and column 33, lines 38-46).

With respect to claim 20, Grey discloses that said computer program contains a plurality of variation points and a plurality of process modification software modules are provided, each of the plurality of process modification software modules comprising at least one user-defined variation function and wherein each of the plurality of variation points is associated with one of the at least one user-defined variation functions (column 13, lines 16-25 and 32-44, column 14, lines 52-65, column 18, lines 28-37 and 49-54, and column 33, lines 38-46).

With respect to claim 21, Grey discloses a computer readable medium containing program instructions which, when executed on a computer, control a measurement process, said instructions comprising (column 11, lines 41-56): an executable variation of the measurement process (column 33, lines 38-46 and column 37, lines 10-19), comprising: a first plurality of instructions generated by a designer of the program of instructions and operable to initiate the measurement process (column 12, line 41 to column 13, line 5); and a second plurality of instructions generated by the designer and operable to control the measurement process (column 5, lines 53-57 and column 7, lines 18-30), the second plurality of instructions including a function call instruction at a variation point (column 20, lines 58-65), the function call instruction being operable to pass control to a user- defined variation function generated by a user (column 20, lines 58-65 and column 33, lines 38-46), the user being different from the designer of the program (i.e. “sequence developer” different from the “user”) (column 20, lines 58-65); wherein the function call instruction passes control to the user-defined variation function when the variation point in the computer

program is reached (column 18, lines 4-17 and column 20, lines 58-65) wherein the user- defined variation function is associated with the function call instruction prior to execution of the measurement process (column 17, lines 25-29, column 18, lines 4-17, and column 20, lines 58-65) and wherein the user-defined variation function operates to modify the measurement process and return control to the measurement process (column 18, lines 4-17, column 20, lines 58-65 and column 33, lines 38-46) and wherein the user is prevented from modifying the measurement process other than through the user-defined variation function (column 32, lines 14-16 and column 34, lines 15-40).

With respect to claim 22, Grey discloses that the function call instruction is operable to pass parameters to the user-defined variation function (column 14, lines 37-50).

With respect to claim 23, Grey discloses that the parameters comprise measurement data (column 14, lines 37-50).

With respect to claim 24, Grey discloses that the function call instruction is operable to receive parameters from the user-defined variation function (column 15, lines 11-14).

With respect to claim 25, Grey discloses that the parameters comprise control parameters, operable to select between a plurality of alternative instructions controlling the measurement process (column 19, lines 33-39 and column 21, lines 5-11).

With respect to claim 26, Grey discloses that the parameters comprise numerical parameters, operable to modify the measurement process (i.e. voltages) (column 30, lines 49-52 and column 46, lines 30-35).

With respect to claim 27, Grey discloses that said measurement process is applied to a device under test and wherein the parameters comprise control codes, operable to cause signals to be supplied to the device under test (column 10, line 62 to column 11, line 6 and column 19, line 64 to column 20, line 5).

With respect to claim 28, Grey discloses that said measurement process is applied to a device under test and wherein the parameters comprise control codes, operable to alter the configuration of the device under test (column 18, lines 62-63 and column 19, lines 33-39).

With respect to claim 29, Grey discloses that the function call instruction invokes an interface (column 12, lines 41-47).

With respect to claim 31, Grey discloses that the user-defined variation function provided by the user of the measurement process is accessed via an interface (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

With respect to claim 32, Grey discloses that said interface operates according to a predetermined protocol (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

With respect to claim 33, Grey discloses that said predetermined protocol is specified at a binary level (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

With respect to claim 36, Grey discloses that said interface is determined by the user and wherein said instructions further comprise instructions to identify the interface (column 12, lines 41-48, column 13, lines 50-62, and column 18, lines 4-16 and 28-37).

With respect to claim 37, Grey discloses that the user-defined variation function is implemented as one of a software component specification for distributed applications or dynamically linked library (i.e. C, C++, JAVA, Visual Basic) (column 13, lines 53-57 and column 14, lines 66-67).

With respect to claim 38, Grey discloses that the second plurality of instructions includes a plurality of function call instructions passing control to a plurality of user-generated variation functions (column 13, lines 16-25 and 32-44, column 14, lines 52-65, column 18, lines 28-37 and 49-54, and column 33, lines 38-46).

With respect to claim 39, Grey discloses that said function call instruction is placed within second plurality of instructions at a variation point where the designer of the instruction program has anticipated that the user may want to interact with or modify the measurement process (column 20, lines 58-65, column 32, lines 14-16, column 33, lines 38-46)

With respect to claim 40, Grey discloses a measurement system comprising a physical interface operable to supply signals to a device under test and receive signals from a device under test (Grey et al.; column 10, line 51 to column 11, line 34).

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 5, 6, 10-13, 34, and 35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Grey in view of U.S. Patent Application Publication No. 2002/0026514 to Ellis et al.

As noted above, the invention of Grey teaches many of the features of the claimed invention and while the invention of Grey does teach connecting the process-modifying host computer to a plurality of specific test instruments (Grey et al., Figure 1), Grey does not specifically indicate that the measurement and process modification be carried out using two separate computers communicating using a Simple Object Access Protocol or Common Object Request Broker Architecture protocol.

Ellis teaches automated tool management in a multi-protocol environment comprising measuring/polling software located on a server computer system with corresponding processor and memory (0025, lines 1-33) and user process control software (0007, lines 1-16) located on a separate remote computer (0023, lines 1-18), wherein the process control software and the monitoring/polling software communicate over a network using predetermined protocol including Common

Object Request Broker Architecture and Simple Object Access Protocol (0007, lines 1-16).

It would have been obvious to one having ordinary skill in the art to modify the invention of Grey to include specifying that the measurement and process modification be carried out using two separate computers communicating using a Simple Object Access Protocol or Common Object Request Broker Architecture protocol, as taught by Ellis, because, as suggested by Ellis, the combination would have provided improved analysis and control of the system of Grey by allowing input and diagnostics by a larger variety of users through remote access while reducing the burden of a user to be local to a UUT during testing (0005, lines 1-33 and 0008, lines 1-14).

Response to Arguments

6. Applicant's arguments filed September 25, 2008, have been fully considered but they are not persuasive.

Applicant argues:

> Rewriting the claim language to correspond to the Examiner's current interpretation of Grey, claims 1 and 21 would require that the sequence developer be prevented from modifying the measurement process other than through the sequence developer-defined variation function, said sequence developer being different from the user of the computer program. The Examiner points to column 32, lines 14-16 and column 34, lines 15-40 as providing this teaching.
>
> The first passage cited teaches that the sequence developer is prevented from modifying a module call, (which specifies which module adapter (LabVIEW, C/CVI etc) is called if the user enables a particular checkbox in the Disable Properties tab. Hence, to the extent that the specification of a module adapter may be construed as part of the measurement process, the first passage may be

taken as teaching that the sequence developer may be prevented from modifying the measurement process. However, the passage does not teach that the sequence developer is nevertheless **allowed to modify** the specification of a module adapter **through the sequence developer defined variation function**.

The second passage cited by the Examiner teaches that the sequence developer may be prevented from modifying settings of other built-in instance properties in individual steps if the user checks the other corresponding checkboxes in the Disable Properties tab. Hence, to the extent that the settings of built-in instance properties may be construed as part of the measurement process, the second passage may also be taken as teaching that the sequence developer may be prevented from modifying the measurement process. However, the passage does not teach that the sequence developer is nevertheless allowed **to modify** these settings **through the sequence developer defined variation function**. Hence, Applicant submits that Grey does not anticipate claims 1, 21, and the claims dependent therefrom.

The Examiner asserts that both the first and second passages discussed above describe the ability of a designer/user to prevent the user/sequence developer from modifying the measurement process by preventing the user from modifying module calls and from modifying property values, specifically:

If the user wants to prevent the sequence developer from modifying the call, he/she enables the Specify Module checkbox in the Disable Properties tab. (column 32, lines 14-16)

The user can use the Disable Properties tab to prevent the sequence developer from modifying the settings of built-in instance properties in individual steps. In this way, the user can make the settings the user specifies in the Step Type Properties dialog box permanent for all step instances.

The tab contains a list of checkboxes. Each checkbox represents one built-in instance property or a group of built-in instance properties. When the user enables the checkbox, the user prevents the sequence developer from modifying the value of the corresponding property or group of properties. FIG. 17 shows the Disable Properties tab of the Step Type Properties dialog box for the Numeric Limit Test step type. Most of the checkboxes in the Disable Properties tab apply to a specific control in the Step Properties dialog box. The two exceptions are the Specify Module checkbox and the Preconditions checkbox.

Specify Module--If the user checks this checkbox, the sequence developer cannot display the Specify Module dialog box on any steps that use the step

type. Check the Specify Module checkbox for step types that always make the same module call.

If the user unchecks the Specify Module checkbox but check the Edit Module Prototype checkbox, the sequence developer can display the Specify Module dialog box but cannot modify any of the parameter information in it. Precondition--If the user checks this checkbox, the sequence developer cannot create preconditions for steps that use the step type. (column 34, lines 15-40)

The Examiner further asserts that these sections were not relied upon for disclosing the ability of the sequence developer to modify the settings through the sequence developer defined variation function as Grey additionally teaches providing a user-generated process modification software module comprising the user-defined variation function for causing the variation, specifically:

By default, each main sequence that is executed uses the process model selected by the user for the entire test station. TestStand has a mechanism called a model callback that allows the sequence developer to customize the behavior of a process model for each main sequence that uses it. By defining one or more model callbacks in a process model, the user specifies the set of process model operations that the sequence developer can customize. (column 20, lines 58-65)

The sequence developer can invoke the Edit substep by selecting a menu item that appears above the Specify Module item in the context menu for the step. Usually, the Edit substep displays a dialog box in which the sequence developer edits the values of custom step properties. For example, an Edit substep might display a dialog box in which the sequence developer specifies the high and low limits for a test. The Edit substep might then store the high- and low-limit values as step properties. (column 33, lines 38-46)

In a template code module for the DLL Flexible Prototype Adapter, the user can access step properties and sequence variables through the TestStand ActiveX API or as parameters to the code module. If the user accesses them as parameters, the sequence developer must specify the parameter values in the Module tab. The values that the sequence developer must specify are usually the same for most step instances. The user can specify default parameter values that TestStand inserts in the Module tab automatically when the sequence developer clicks on the Create Code button. (column 37, lines 10-19)

Applicant argues:

> Claims 2 and 31 depend from claims 1 and 21 respectively and further require that the process modification software module further comprise an interface servicing element that services an interface realized by the measurement process. The Examiner points to column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17 for the additional teachings.
>
> The first passage cited teaches that the external code modules identified by the Examiner as process modification software modules are steps within sequences. Figure 2, associated with the first cited passage shows that the sequence files/external code modules are called up through the adapters 240. The only interface servicing element discussed in the passage is the adapter interface 232, which Figure 2 shows as the element linking the TestStand Engine 220 and the adapters 240. Hence, interface 232 is clearly not part of the sequence files/external code modules.

Claim 2 requires "wherein the process modification software module further comprises an interface servicing element that services an interface realized by the measurement process". The Examiner asserts that the term "interface" is broadly interpreted to be some type of program and/or code designed to communicate information and the term "realized' is broadly interpreted to mean recognized. As such, Grey discloses an interface servicing element that services an interface realized by the measurement process, specifically by disclosing a servicing element that provides code for communicating information as part of the TestStand Engine, sequences, code modules, and/or module adapters which are all recognized by the measurement process, specifically:

> As shown in FIG. 2, the TestStand Engine 220 plays a pivotal role in the TestStand architecture. The TestStand Engine 220 runs sequences. Sequences contain steps that can call external code modules. By using module adapters 240 that have the standard adapter interface 232, the TestStand Engine 220 can load and execute different types of code modules. (column 12, lines 41-48)

> Most steps in a TestStand sequence invoke code in another sequence or in a code module. When invoking code in a code module, TestStand must know the type of the code module, how to call it, and how to pass parameters to it. The different types of code modules include LabVIEW VIs, C functions in DLLs, and C functions in source, object, or library modules that are created in LabWindows/CVI or other compilers. TestStand also must know the list of parameters that the code module requires.
>
> In the preferred embodiment, TestStand uses module adapters 240 to obtain this knowledge. In the currently implemented embodiment, TestStand provides the following module adapters: (column 13, lines 50-62)

> A value is a number, a string, or a Boolean. TestStand stores numbers as 64-bit floating-point values in the IEEE 754 format. (column 15, lines 15-17)

With respect to claim 31, the Examiner further asserts that the parameters provided by the user as part of the user-defined variation function is accessed by the TestStand Engine, sequences, code modules, and/or module adapters and, consequently, by their interfaces and therefore Grey discloses that the user-defined variation function provided by the user of the measurement process is accessed via an interface.

Applicant argues:

> The second passage cited by the Examiner discusses the module adapters 240, suggesting that the Examiner may be identifying these as the interface

servicing elements recited by the claims. However, Figure 2 also shows that adapters 240 may connect to the sequence files, but are not part of those files.

As noted above, using a broadest reasonable interpretation of the term "interface", the Examiner maintains that Grey discloses that the process modification software module further comprises an interface servicing element that services an interface realized by the measurement process

Applicant argues:

Applicant is unsure of the relevance of the third cited passage, as it does not appear to concern either the sequence files or any interface servicing element. Hence, Applicant submits that there are additional grounds for allowing claims 2, 31 and the claims dependent therefrom.

The Examiner asserts that the third passage describes that the values entered by the user as part of the user-defined variation function are either in terms of numbers, strings, or Booleans which are stored as 64-bit floating-point values in the IEEE 754 format. Therefore, the Examiner asserts that Grey discloses that the interfaces implemented in obtaining and transmitting data and executing the measurement process using the values of the user-defined variation function operate according to a predetermined protocol specified at a binary protocol as Booleans are logical datatypes specified in terms of a binary true/false.

Applicant argues:

Claims 3 and 32 depend from claims 2 and 22 respectively, and further require that said interface operates in accordance with a predetermined protocol.

> The Examiner points to column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15- 17 as providing the additional teaching. At most, the cited passages concern different types of code modules and corresponding compilers but do not teach predetermined protocols for interfaces. Hence, there are additional grounds for allowing claims 3, 32 and the claims dependent therefrom.

For the reasons provided above, the Examiner maintains that Grey discloses that said interface operates in accordance with a predetermined protocol (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

Applicant argues:

> Claims 4 and 33 depend from claims 3 and 32 respectively and further require that said predetermined protocol is specified at a binary level. The Examiner points to column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17 for the additional teaching. At most, the third cited passage teaches that binary values may be used as properties in the expressions used within the code modules. Hence, there are additional grounds for allowing claims 4 and 33.

For the reasons provided above, the Examiner maintains that Grey discloses that said predetermined protocol is specified at a binary level (column 12, lines 41-48, column 13, lines 50-62, and column 15, lines 15-17).

Applicant argues:

> Claim 7 depends from claim 2 and further requires that the interface has an identity which is determined by, according to the Examiner's interpretation, the sequence developer, and passed into the measurement process. The Examiner points to column 12, lines 41-48, column 13, lines 50-62, and column 18, lines 4-16 and 28-37 for the additional teaching. At most, the cited passages teach that the identity of the type of external code module may be passed to the TestStand Engine using module adapters 240. This is not equivalent to teaching that the identity of any interface is passed into the measurement process. Hence, there are additional grounds for allowing claim 7.

> Claim 36, which depends from claim 21, has additional requirements similar to those of claim 7 regarding the identity of the interface. The Examiner points to the same passages in Grey discussed above with respect to claim 7. Applicant must repeat the arguments presented above, that the teachings regarding the external code module are not equivalent to teachings regarding any interface. Hence, there are additional grounds for allowing claim 36.

As discussed above, the Examiner asserts that the term "interface" is broadly interpreted to be some type of program and/or code designed to communicate information. As such, the Examiner asserts that Grey discloses code that has a specific condition (i.e. identity) which is determined by the user and is identified and passed into said measurement process, specifically, by generating and passing sequence developer created steps with corresponding parameters into the measurement process:

> As shown in FIG. 2, the TestStand Engine 220 plays a pivotal role in the TestStand architecture. The TestStand Engine 220 runs sequences. Sequences contain steps that can call external code modules. By using module adapters 240 that have the standard adapter interface 232, the TestStand Engine 220 can load and execute different types of code modules. (column 12, lines 41-48)

> Most steps in a TestStand sequence invoke code in another sequence or in a code module. When invoking code in a code module, TestStand must know the type of the code module, how to call it, and how to pass parameters to it. The different types of code modules include LabVIEW VIs, C functions in DLLs, and C functions in source, object, or library modules that are created in LabWindows/CVI or other compilers. TestStand also must know the list of parameters that the code module requires.
> In the preferred embodiment, TestStand uses module adapters 240 to obtain this knowledge. In the currently implemented embodiment, TestStand provides the following module adapters: (column 13, lines 50-62)

> The sequence developer creates a new step by selecting the "Insert Step" item in the context menu that appears when the user right click on a sequence window. The "Insert Step" item brings up a hierarchical submenu containing the

step types available on the computer. When the user creates a new step type, the user specifies its name and position within the submenu.

Source Code Templates

When the user creates a step type, the user can also define source code templates for that step type. When the sequence developer creates a new step of that type, the developer can use a source code template to generate source code for the step module.

…

Each sequence has its own list of parameters. The user can specify the number of parameters and the data type of each parameter. The user can also specify a default value for each parameter. When the sequence developer creates a step that calls one sequence from another, the developer can specify the values to pass for the parameters of the subsequence. If the developer does not specify the value of a parameter, the TestStand Engine 220 passes the default value. The user can use the TestStand ActiveX API to access sequence parameter values from code modules that steps in the sequence call. (column 18, lines 4-16 and 28-37)

Applicant argues:

Claims 19 and 20 depend from claim 1 and further require that each of a plurality of variation points in the computer program be associated with one of a plurality of user-defined functions in the process modification software module. The Examiner points to column 13, lines 16-25 and 32-44, column 14, lines 52-65, column 18, lines 28-37 and 49-54, and column 33, lines 38-46 as providing the additional teachings. The cited passages teach the possibility of multiple concurrent executions of a sequence, of setting breakpoints, of calling the Engine API from a plurality of test modules, of accessing variables, of calling one sequence from another, and of specifying step properties. However, there is no teaching in any of the cited passages regarding the association of each of a plurality of variation points with one of a plurality of user-defined functions, as required by the claims. Hence, Applicant submits that there are additional grounds for allowing claims 19 and 20.

Claim 38, which depends from claim 21, has additional requirements similar to those of claims 19 and 20 regarding a plurality of function call instructions passing control to a plurality of user-generated variation functions. The Examiner points to the same passages in Grey discussed above with respect to claims 19 and 20. Applicant submits that there is no teaching in the cited passages regarding a plurality of function call instructions passing control to a plurality of

user-generated variation functions. Hence, there are additional grounds for allowing claim 38.

The Examiner asserts that the cited passages indicate that the user has complete flexibility in controlling the measurement process and the user is allowed to obtain values of variables, properties, and new expressions periodically throughout the measurement process. These passages additionally disclose the ability of the user to provide opportunities for the sequence developer to create a plurality of additional steps as part of the measurement variation at any of a plurality of points in the process. As such, the measurement process provides a plurality of locations (i.e. variation points) where the sequence developer interacts with the process modification to provide the user-defined functions and therefore Grey discloses that the computer program contains a plurality of variation points and that the process modification software module comprises a plurality of user-defined functions, wherein each of the plurality of variation points is associated with one of the plurality of user-defined functions, specifically:

> Although the user can use the TestStand sequence editor 212 at a production station, the TestStand run-time operator interfaces 202 are simpler and are fully customizable. Like the sequence editor 212, the run-time operator interfaces 202 allow the user to start multiple concurrent executions, set breakpoints, and single step. Unlike, the sequence editor 212, however, in the present embodiment the run-time operator interfaces 202 do not allow the user to modify sequences, and they do not display sequence variables, sequence parameters, step properties, and so on.
>
> ...
>
> The TestStand Test Executive Engine 220 is used for creating, editing, executing, and debugging sequences. In the preferred embodiment, the TestStand Test Executive Engine 220 comprises a set of DLLs that export an object-based or component-based API, preferably an ActiveX Automation API.

The TestStand sequence editor 212 and run-time operator interfaces 202 use the TestStand Test Executive Engine API (Engine API). The user can call the Engine API from any programming environment that supports access to ActiveX Automation servers. Thus, the user can call the Engine API from test modules, including test modules that are written in LabVIEW and LabWindows/CVI. (column 13, lines 16-25 and 32-44)

In TestStand, the values of variables and properties can be used in numerous ways, such as passing a variable to a code module or using a property value to determine whether to execute a step. Sometimes the user desires to use an expression, which is a formula that calculates a new value from the values of multiple variable or properties. An expression can be used anywhere a simple variable or property value is used. In expressions, the user can access all variables and properties in the sequence context that is active when TestStand evaluates the expression. The following is an example of an expression:

Locals.MidBandFrequency=(Step.HighFrequency+Step.LowFrequency)/2

(column 14, lines 52-65)

The sequence developer creates a new step by selecting the "Insert Step" item in the context menu that appears when the user right click on a sequence window. The "Insert Step" item brings up a hierarchical submenu containing the step types available on the computer. When the user creates a new step type, the user specifies its name and position within the submenu.

Source Code Templates

When the user creates a step type, the user can also define source code templates for that step type. When the sequence developer creates a new step of that type, the developer can use a source code template to generate source code for the step module.

...

Each sequence has its own list of parameters. The user can specify the number of parameters and the data type of each parameter. The user can also specify a default value for each parameter. When the sequence developer creates a step that calls one sequence from another, the developer can specify the values to pass for the parameters of the subsequence. If the developer does not specify the value of a parameter, the TestStand Engine 220 passes the default value. The user can use the TestStand ActiveX API to access sequence parameter values from code modules that steps in the sequence call. (column 18, lines 4-16 and 28-37)

> The sequence developer can invoke the Edit substep by selecting a menu item that appears above the Specify Module item in the context menu for the step. Usually, the Edit substep displays a dialog box in which the sequence developer edits the values of custom step properties. For example, an Edit substep might display a dialog box in which the sequence developer specifies the high and low limits for a test. The Edit substep might then store the high- and low-limit values as step properties. (column 33, lines 38-46)

Additionally, the Examiner asserts that one having ordinary skill in the art would recognize that it is extremely common in program generation, such as the program generation in Grey, that the program designer has the ability to provide a plurality of individual points in the program where interaction with the user is desired.

Further, since the invention of Grey discloses associating a function call with a user-generated variation function and also teaches a plurality of user-generated variation functions as desired at a plurality of variation points, the Examiner maintains that Grey discloses a plurality of function call instructions passing control to a plurality of user-generated variation functions

Conclusion

7. The prior art made of record and not relied upon is considered pertinent to Applicant's disclosure.

U.S. Patent No. 6,907,557 to Perez et al. (incorporating by reference U.S. Patent No. 6,401,220 to Grey et al.) teaches a system and method for testing a group of related products wherein a user is permitted to modify the measurement process by configuring parameters (Perez et al.; column 4, lines 49-63 and column 10, line 57 to column 11, line 14), such as the parameters used through the user-defined variation

function (Grey et al.; column 14, lines 52-65), while preventing the user from modifying the measurement process through particular sequences (Perez et al.; column 4, lines 49-63 and column 10, line 57 to column 11, line 14).

U.S. Patent No. 6,449,741 to Organ et al. teaches a single platform electronic tester comprising means for controlling testing of a DUT (column 4, lines 26-34) using a program executed by a user (column 4, lines 45-55) wherein the user is allowed to arrange the flow of test execution (column 4, lines 56-64) for performing measurements (column 6, lines 29-32) while the operator is allowed to selectively control modification of the test by preventing the user from modifying the test/measurement process/program (column 13, lines 30-32 and column 14, lines 13-17).

U.S. Patent No. 6,308,326 to Murphy et al. teaches run-time modules for dynamically adjusting computer operation.

U.S. Patent No. 6,769,114 to Leung teaches methods and apparatus for preventing software modification from invalidating previously passed integration tests.

U.S. Patent Application Publication No. 2003/0046665 to Ilin teaches a reusable software component for textually supplementing, modifying, evaluating, and processing procedural logic for a compiled host program at run-time.

U.S. Patent No. 6,766,514 to Moore teaches a compiler having real-time tuning, I/O scaling and process test capability.

U.S. Patent No. 6,351,843 to Berkley et al. teaches dynamically inserting a function into an application executable at runtime.

U.S. Patent No. 6,202,043 to Devoino et al. teaches a computer based system for imaging and analyzing a process system and indicating values of specific design changes.

U.S. Patent No. 6,163,879 to Mackey teaches an interface and method for facilitating writing and modifying of lines of programming code.

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to JEFFREY R. WEST whose telephone number is

(571)272-2226. The examiner can normally be reached on Monday through Friday, 8:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eliseo Ramos-Feliciano can be reached on (571)272-7925. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Jeffrey R. West/
Primary Examiner, Art Unit 2857

January 6, 2009